IMS GLC IWB/CFF Specification

Version 1.0 Final Specification

Date Issued: 1 February 2012
Latest version: http://www.imsglobal.org/iwbcff/

IPR and Distribution Notices

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imsglobal.org/ipr/imsipr_policyFinal.pdf.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: http://www.imsglobal.org/speclicense.html.

Permission is granted to all parties to use excerpts from this document as needed in producing requests for proposals.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

Join the discussion and post comments in the IWB/CFF Public Forum:
http://www.imsglobal.org/community/forum/categories.cfm?catid=145&entercat=y

Table of Contents

1 Introduction

The Interactive WhiteBoard/Common File Format (IWB/CFF) specification defines a file format to hold content primarily designed to be viewed on a large display. Much of this content will be designed to be interactive, so objects can move around the page.

The primary goal of this format is to establish a format that can be opened, edited, saved and used across many whiteboard applications so that teaching content can be exchanged between establishments. To this goal the format must be simple but extendible in a restricted way to ensure compatibility.

The format is called the "Interactive Whiteboard File Format" or IWB for short.

1.1 Nomenclature

File	This is the actual file on a computer.
Document	This is the collection of one or more pages available in a file.
Page	This is where any elements are placed.
Slide	This is the area on a page that should primarily be shown on a whiteboard, it is defined with the viewbox attribute.
Elements	These are the objects which appear on a page.
Reader	The application that is opening the file to display it.
Writer	The application that is saving the file, either the first creation or by resaving an existing file.
SVG	Scalar Vector Graphics format.
SVGT	SVG Tiny specification.
Paint server	This is an SVG term for filling objects with things such as gradients or patterns. The IWB format only allows filling with a solid color.

1.2 References

XML 1.0	http://www.w3.org/TR/REC-xml/
SVG 1.1	http://www.w3.org/TR/SVG11/
SVG 1.2 (draft)	http://www.w3.org/TR/2004/WD-SVG12-20041027/
SVG Tiny 1.2 (draft)	http://www.w3.org/TR/SVGMobile12/
CSS2	http://www.w3.org/TR/REC-CSS2/cover.html

2 File Overview

This file uses the Extensible Mark-up Language (XML) 1.0 specification.

All IWB files shall have an extension of "IWB" in their name (e.g. "afile.iwb"). The IWB file shall be a zip file. If the content of a IWB file can be represented entirely by XML, the zip file shall contain only a single XML file named "content.xml".

The zip file can store multiple files inside itself and is used when other media is needed, for example images. The zip compressed file will also contain a single text file named "content.xml" describing the IWB document.

2.1 IWB ZIP File

The zipped file must contain an xml file describing the document and CAN have several folders. There is a folder for each media, a folder to hold a thumbnail of the document and a folder to hold additional files. The names are:

File: "content.xml" – Main contents of file.

Folder: "images" – Need only exist if images are used in the file.

Folder: "videos" – Need only exist if videos are used in the file.

Folder: "audio" – Need only exist if audios are used in the file.

Folder: "flash" – Need only exist if flash is used in the file.

Folder: "thumbnails" – Need only exist if a thumbnail has been created.

Folder: "additional" – Need only exist if additional files are included.

Sub folders can be added to the media folders as this may be useful for documents which include flash, or to store a large amount of media. The actual media files must be given unique names to avoid file name clashes of files residing in the same folder and they must preserve the extension on the filename so that the file format is easily determined.

The thumbnail folder contains an optional image of the key page in the document; this will almost certainly be an image of the first page but does not need to be. The file name should begin "thumbnail" and preferably be a Portable Network Graphics file or a Bitmap file, i.e. "thumbnail.png". This is primarily to give applications and operating systems the ability to show a preview of the file.

2.2 The XML File

The format uses the Scalar Vector Graphic format (SVG) to represent much of the content, with additional attributes outside of the SVG specification being added as IWB specific tags. Much of the SVG parts have been simplified from the full specification to make the IWB format simpler to implement, however those specific tags and attributes included in this format follow the specifications set down in the SVG standard unless additional rules explicitly mention otherwise.

The SVG part of the file is self contained between the <svg:svg> tags, the IWB meta tags appear before the SVG portion and the additional IWB properties after.

XML Comments can appear at any point in the file and are marked by a "<!--" and "-->". The comments should be ignored by a Reader application and only exist to help reading of the file by hand.

All IWB files document have a typical order. The main xml declaration and iwb namespace and schema location declarations come first. These are followed by optional meta data tags. The SVG section comes next followed by the final iwb tags section. This shows a typical file structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<iwb version="1.0" xmlns="http://www.imsglobal.org/xsd/iwb_v1p0"
    xmlns:svg="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.imsglobal.org/xsd/iwb_v1p0
```

```
http://www.imsglobal.org/profile/iwb/iwbv1p0_v1p0.xsd http://www.w3.org/2000/svg
http://www.imsglobal.org/profile/iwb/svgsubsetv1p0_v1p0.xsd http://www.w3.org/1999/xlink
http://www.imsglobal.org/xsd/w3/1999/xlink.xsd">
    <!-- Meta data describing file -->
    <meta content="Klaus Meine" name="owner"/>
    <svg:svg viewbox="0 0 1000 760">
        <!-- Contents of document, the svg tags -->
        <svg:image height="760" id="background" width="756" x="122"
            xlink:href="images/mybackground.png" y="0"/>
        <svg:a id="link1" xlink:href="http://creativecommons.org/licenses/by-nc-sa/2.0/uk/">
            <svg:image height="31" width="88" x="900"
                xlink:href="images/CC-logo-88x31.png" y="700"/>
        </svg:a>
    </svg:svg>
    <!-- Additional properties of elements and other iwb tags -->
    <element background="true" ref="background"/>
    <link file="external" ref="link1"/>
</iwb>
```

2.2.1 Namespaces

An IWB file as defined in this specification depends upon XML elements and attributes from different XML schemas produced by different organizations. Each individual schema is defined with its own target namespace. Therefore, these individual namespaces are exposed in IWB instance documents such as the example shown above in section 2.2. The use of the xsi:schemaLocation attribute declares every namespace used by the IWB specification and also the location of every XML Schema document used. This allows specification implementers easy access to all of the XML Schema and embedded Schemtron rules employed to measure compliance with the specification.

2.3 Meta Section

The meta data section will hold several pieces of information:

owner – The name of the person who created it.

description – A description of what the file holds.

creator – The name and version of the application in which the file was created.

The owner and description are optional and will depend on whether the Writer application has functionality to allow these to be set. The creator is also optional but should be added in most cases. The metadata is added by a single tag. There are two attributes, the first determines which data it is and the second is the content of that data:

```
<meta name="description" content="A description of the file"/>
```

Note: In version 2.0 of this specification, the meta data tag will be extended to include full IMS metadata.

A basic file with a locked red rectangle in the center looks like this: (for the sake of brevity, required namespace and schema location declarations are omitted from the remaining examples)

```
<?xml version="1.0" encoding="UTF-8"?>
<iwb>
    <meta name="owner" content="BECTA"/>
    <meta name="description" content="A little red box"/>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" fill="red" x="450" y="450" width="100" height="100" />
    </svg:svg>
    <element ref="rect1" locked="true" />
</iwb>
```

3 Document

3.1 Pages

A document has one or more pages. To define multiple pages you use the SVG tags <svg:pageset> and <svg:page> , if these are not used then the document is assumed to have a single page holding all its elements. If they are used then all elements must appear inside a <svg:page> tag.

This is an example of a document with two pages, the first has a red rectangle and the second has a green rectangle:

```
<?xml version="1.0" encoding="UTF-8"?>
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
    <svg:pageset>
        <svg:page id="page1">
            <svg:rect id="rect1" fill="red" x="450" y="450"
            width="100" height="100"/>
        </svg:page>
        <svg:page id="page2">
            <svg:rect id="rect2" fill="green" x="450" y="450"
            width="100" height="100"/>
        </svg:page>
    </svg:pageset>
    </svg:svg>
</iwb>
```

A page has an id so that navigation between pages can be achieved. For instance, an element on one page can be linked to another page. See section “11 Links”. Backgrounds can be set for each page, see section “10 Background”.

The order of the pages is defined by the place they appear in the file; the first page is the one highest in the file.

Each page has its own slide area but this is only defined once by the viewbox on the <svg:svg> element, see “3.2 Positioning”.

Each page may hold any number of elements.

3.2 Positioning

The documents need to look the same across many different screen resolutions, therefore the positioning in any documents must be a relative one.

This format uses the viewbox concept from SVG as a starting point. The viewbox values are in user co-ordinates and they define how all the elements are positioned.

As in SVG the viewbox is defined with a start position and a width and height separated by spaces or commas. The (0,0) position is defined in the top left corner with x values increasing across the page to the right and y values increasing down the page. For instance, a viewbox with a starting position of (0,0) , a width of 200 and a height of 100 would be written as:

```
<svg:svg viewbox="0 0 200 100">
```

The top edge of a page is either at the zero y position or at the viewbox y position if the viewbox y position is less than zero. The bottom edge is either at the viewbox y position plus its height or at the bottom of the lowest element if this is beyond the viewbox. The left edge of the page is either at the zero x position or at the viewbox x position if the bottom of the lowest element is less than zero. The right edge is either at the viewbox x position plus its width or at the right of the right most element if the right of the right most element is beyond the viewbox.

As the user co-ordinates do not necessarily have a direct relation to an actual screen size or to an aspect ratio we need a way to record this. So in addition to the viewbox we can also include the width and height of the resolution the file was created / edited on. If these are omitted then the viewbox must match the aspect ratio of the screen measurement it was last saved on.

Although in SVG the width and height can be defined in different measurements (such as Inches or Centimetres) in the IWB these values are always defined in pixels.

These are equivalent:

```
<svg:svg width="800" height="600" viewbox="0 0 1000 1000">

<svg:svg width="800px" height="600px" viewbox="0 0 1000 1000">
```

But this is an error:

```
<svg:svg width="5cm" height="3cm" viewbox="0 0 1000 1000">
```

This example shows a rectangle in the center of the screen. The file was originally saved on a screen with a resolution of 800 by 600.

```
<?xml version="1.0" encoding="UTF-8"?>
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect x="450" y="450" width="100" height="100">
    </svg:svg>
</iwb>
```

Note that the width and height is the area of the screen used by a slide and not necessarily the whole screen. For instance if part of the screen is covered by some border and the slide appears inside this border then the width and height are the values inside the border. For example, on a screen with a resolution of 800 by 600, if there's a border that covers 10 pixels on each screen edge, the setting would be:

```
<svg:svg width="780" height="580" viewbox="0 0 1000 1000">
```

In IWB the viewbox defines the area of a page which should be shown as the "slide". Elements can exist outside the viewbox and these should not be visible on the slide, but can exist in the document (these could be viewed for example by scrolling, or opening a notes section but this depends on the abilities of the Reader). Elements that are partly inside the viewbox should be assumed to be partly visible on the slide.

In this example the slide area shows the rectangle with id "rect1" in the centercenter and the top half of the rectangle with id "rect2" just below it. If scrolling on the Reader is enabled then the other half of the "rect2" is visible. The rectangle "rect3" does not appear on the slide at all but still exists in the document below the slide area.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" x="450" y="450"
            width="100" height="100"/>
        <svg:rect id="rect2" x="450" y="950"
            width="100" height="100"/>
        <svg:rect id="rect3" x="450" y="1200"
            width="100" height="100"/>
    </svg:svg>
</iwb>
```

3.3 Container

The container tag <svg:g> can be used to set default attribute values across several elements at once instead of placing values in each element. It is also used to create sections in the switch statement, see section "12 Fall back".

To set a list of elements to be filled with red with a black stroke you could embed the objects in the <svg:g> tag and set the attributes there:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:g fill="red" stroke="black">
            <svg:rect x="1" y="1" width="10" height="10"/>
            <svg:circle cx="10" cy="20" r="5"/>
            <svg:ellipse cx="10" cy="30" rx="5" ry="10"/>
        </svg:g>
    </svg:svg>
</iwb>
```

The <svg:g> elements can have further <svg:g> elements inside themselves. So here the circle is drawn with a yellow stroke but the ellipse uses the default black stroke:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:g fill="red" stroke="black">
```

```
            <svg:g stroke="yellow">
                <svg:circle cx="10" cy="20" r="5"/>
            </svg:g>
            <svg:ellipse cx="10" cy="30" rx="5" ry="10"/>
        </svg:g>
    </svg:svg>
</iwb>
```

The <svg:g> tags should only exist inside the SVG part of the file. The tag should not be confused with the <iwb:group> tag which joins elements together. The <svg:g> tag has no effect on the elements other than setting styles.

4 Elements: All

There are three different types of elements:

- Shapes
- Text
- Media

Elements are the objects that appear on a page. Each element is positioned relative to the viewbox with an x and y co-ordinate, using the same user co-ordinates. Any element off the edge of a page is treated as an invisible element and these should not be shown anywhere on a page. An element is off the edge of a page if all parts of the object, after transforms, have co-ordinates which are more negative than the top or left hand edge of the current page (see section “3.1 Positioning”).

The elements are mainly based on SVG standards but have additional properties set through a <iwb:element> tag. This tag uses a reference to match up with SVG elements. Any element which has IWB attributes must have an id set in the SVG part of the file. The <iwb:element> tags are located outside and after the svg part towards the end of the file.

For instance this file has two rectangles. One of the rectangles has an id of “rect1”, this is used in the <iwb:element> as ref=”rect1” , the rectangle is then set to be locked. The second rectangle does not have, nor need, an id. It takes the default initial value for locked of “false” (see section “4.3 Locking”).

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" fill="red" x="450" y="450"
            width="100" height="100"/>
        <svg:rect fill="green" x="470" y="470" width="60" height="60"/>
    </svg:svg>
    <element ref="rect1" locked="true"/>
</iwb>
```

The sub chapters here refer to attributes that can be placed on any of the elements.

4.1 Layering Elements

The layering, or z-order, of elements is defined implicitly in the file by the order the objects appear in the SVG part. The first elements are at the bottom of the z-order (behind everything else), those further down are higher in the z-order.

Here a green rectangle appears on top of the red rectangle.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" fill="red" x="450" y="450"
            width="100" height="100"/>
        <svg:rect id="rect2" fill="green" x="470" y="470"
            width="60" height="60"/>
    </svg:svg>
</iwb>
```

4.2 Rotation and Translation of Elements

Rotation is done with the SVG transform attribute. For example, a rectangle can be rotated like so:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" transform="rotate(-30)" fill="red"
              x="450" y="450" width="100" height="100"/>
    </svg:svg>
</iwb>
```

By default rotation occurs around the current coordinates zero position, but the point of rotation can also be supplied as follows:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" transform="rotate(-30, 500, 500)" fill="red"
           x="-50" y="-50" width="100" height="100"/>
    </svg:svg>
</iwb>
```

In the above example the rotation is -30 about the centercenter point 500,500.

Transforms can also appear on the container tag <svg:g>. As here:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:g transform="translate(500,500) rotate(-30)">
            <svg:rect id="rect1" fill="red" x="-50" y="-50"
                      width="100" height="100"/>
        </svg:g>
    </svg:svg>
</iwb>
```

In all cases transforms occur to the co-ordinate system before an element is placed on the page.

4.3 Locking Elements

Locking an element means that it can not be “moved” or “selected” in the Reader by a user, however any links set on the element should still be accessible.

Locking is an attribute of the IWB and is added with the <iwb:element> tag.

locked – whether an element is locked.

Value: [true | false]

Initial: “false”.

Referenced to: Any SVG element.

Here is an example of a locked rectangle.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" fill="red" x="450" y="450"
               width="100" height="100"/>
    </svg:svg>
    <element ref="rect1" locked="true"/>
</iwb>
```

4.4 Replicating Elements

Elements can be set to replicate which means that when a user tries to move the element, a copy of the element is created and this is moved, the original element stays where it is.

Replicating is an attribute of the IWB and is added with the <iwb:element> tag.

replicate – whether an element should produce replicates of itself.

Value: [true | false]

Initial: “false”.

Referenced to: Any SVG element.

Here is an example of a replicating rectangle.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="rect1" fill="red" x="450" y="450"
               width="100" height="100"/>
    </svg:svg>
    <element ref="rect1" replicate="true"/>
</iwb>
```

5 Elements: Shapes

The IWB file stores these SVG elements:

- line
- circle
- ellipse
- rectangle
- polyline (also represents freehand)
- polygon

These fall into two types. Elements composed of one or more lines only (“line” and “polyline”) and elements which can also be filled.

5.1 General Line Properties

The line properties apply to all shapes and can be set with the following SVG attributes, these follow the recommendations in SVG1.0 unless otherwise stated:

stroke – color of graphic outline, or ‘none’ to indicate no line. Unlike SVG, there are no “paint servers”.

stroke-opacity – the level of transparency for the line.

stroke-width – width of the line.

stroke-linecap – how the end of a line will appear.

stroke-linejoin – how line joins will appear.

stroke-dasharray – comma-separated list of dash and gap length.

5.2 Additional Line Properties

There are additional IWB line attributes, which are added with the <iwb:element> tag:

stroke-lineshape-start – This lets you set one of several shapes that will appear at the start of a line, such as an arrow shape.

Values: [none | arrow | circle | line]

Initial: “none”.

Referenced to: svg:line or svg:polyline.

stroke-lineshape-end – This lets you set one of several shapes that will appear at the end of a line, such as an arrow shape.

Values: [none | arrow | circle | line]

Initial: “none”.

Referenced to: svg:line or svg:polyline.

freehand – This signifies that a polyline was actually created by freehand, rather than be individual lines.

Values: [true | false]

Initial: “false”

Referenced to: svg:polyline

highlight – This signifies that a polyline was used to highlight a part of the screen. The highlight attribute implicitly sets a stroke-opacity value if none was set.

Values: [true | false]

Initial: “true”.

Referenced to: svg:polyline.

A red line with an arrow shape at the end would look like this:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:line id="line1" x1="200" y1="200" x2="300" y2="300"
           stroke="red" stroke-width="5"/>
    </svg:svg>
    <element ref="line1" stroke-lineshape-end="arrow"/>
</iwb>
```

5.3 General Fill Properties

The fill properties can be set with the following SVG attributes, these follow the recommendations in SVG1.0 unless otherwise stated:

- **fill** – this is the color of graphic interior, or none to indicate empty. Unlike SVG there are no “paint servers”.
- **fill-opacity** – the level of transparency for the fill.

Also note that unlike SVG, there is no fill-rule attribute in the IWB, all fills should use the evenodd algorithm.

5.4 Additional Shape Properties

These are an additional IWB shape attributes added with the <iwb:element> tag.

revealer – This signifies that a rectangle is specifically used to hide part of a slide so it can be subsequently revealed by a user. It will depend on the Reader how this is to be interpreted.

Values: [true | false]

Initial: “false”

Referenced to: svg:rect

5.5 The Shape Tags

5.5.1 Rectangle

As in SVG the rectangle is defined with an x and y co-ordinate, and a width and height. The IWB format does not recognise the rounding of the corners as is stated in SVG.

```
<svg:rect id="rect1" x="400" y="100" width="400" height="200"
    fill="yellow" stroke="navy" stroke-width="10"/>
```

The rectangle can also be set as a “revealer”, see section “5.4 Additional Shape Properties”.

5.5.2 Circle

As in SVG the circle takes an x and y co-ordinate for its centercenter and a radius r.

```
<svg:circle id="circ1" cx="600" cy="200" r="100" fill="red" stroke="blue"
    stroke-width="10"/>
```

5.5.3 Ellipse

As in SVG the ellipse takes an x and y co-ordinate for its centercenter and two radii rx and ry.

```
<svg:ellipse cx="500" cy="500" rx="250" ry="100" fill="red" stroke="navy"
    stroke-width="10"/>
```

5.5.4 Line

As in SVG the line takes two x and y co-ordinates.

```
<svg:line x1="100" y1="300" x2="300" y2="100" stroke="blue" stroke-
    width="5"/>
```

The line can be set with the additional properties “stroke-lineshape-start”, “stroke-lineshape-end”. See section“5.2 Additional Line Properties”.

5.5.5 Polyline

As in SVG the polyline takes a list of x and y co-ordinates, consisting of move and draw commands.

```
<svg:polyline points="50,375 150,375 150,325 250,325 250,375"
    stroke="blue" stroke-width="10"/>
```

The polyline can be set with the additional properties “stroke-lineshape-start”, “stroke-lineshape-end”, “freehand” and “highlight”. The polyline when used as freehand tool will usual differ only by a greater number of points much closer together than a normal polyline. See section “5.2 Additional line properties”.

5.5.6 Polygon

As in SVG the polygon takes a list of x and y co-ordinates, consisting of move and draw commands.

```
<svg:polygon points="350,75 379,161 469,161 397,215" fill="red"
    stroke="blue" stroke-width="10"/>
```

6 Elements: Text

Text is represented as either a single line with the SVG <svg:text> element or by an area with the SVGT1.2 <svg:textarea> element. The <svg:textarea> element is useful for sections of text which need to be word-wrapped across several lines and for sections or lines of text that need to be justified in some way. The <svg:text> element is useful when more accurate positioning is needed.

Changes to the default text styles inside the <svg:text> and <svg:textarea> elements are controlled with the <svg:tspan> tag. The <svg:tspan> tag is not classed as an element, it is only used inside the text elements.

Here is a text example with a single word changed to red.

```
<svg:text x="10" y="10" font-family="Verdana" font-size="45">
     This is some <svg:tspan fill="red">red </svg:tspan>text.
</svg:text>
```

Here is a textarea which sets the font and size of the text and which wraps text and also forces a newline with the <svg:tbreak> tag.

```
<svg:textarea font-family="Verdana" font-size="45" x="10" y="60"
    width="50" height="90">
    This line will wrap into separate lines. This line
    breaks<svg:tbreak/> in two.
</svg:textarea>
```

All text inside either <svg:text> or <svg:textarea> must be properly escaped either by using escape characters or with an XML CDATA section. This example is incorrect as the “&” and “<” characters should be escaped:

```
<svg:textarea x="0" y="0" width="200" height="100">
    This is incorrect text with an ampersand "&" and
    this is incorrect text with an angle bracket "<".
</svg:textarea>
```

The two different ways to escape characters are shown here:

```
<svg:textarea x="0" y="0" width="200" height="100">
    This is correct text with an ampersand "&amp;" and
    <![CDATA[this is also correct text with an angle bracket "<".]]>
</svg:textarea>
```

6.1 General Text Properties

The text properties can be set inside <svg:text>, <svg:textarea> and <svg:tspan> with the following SVG attributes, these follow the recommendations in SVG1.0 unless stated otherwise:

fill – The color of the text. Unlike SVG there are no “paint servers”.

font-family – The name of font used.

font-size – The size of font used. This is defined in user co-ordinates (set in the viewbox) so that text appears the same size across different Readers.

font-stretch – Stretched horizontally of the text.

font-style – Style of font, italic for instance.

font-weight – How bold the text appears.

The following are attributes from SVG Tiny 1.2

text-align – How the text is justified; e.g. centerd or left-aligned.

Values: [start | end | center | justify]

Initial: “start”.

Referenced to: svg:textarea or svg:tspan in svg:textarea.

Links can also be added to the text with an <svg:a> tag. See section “11 Links”.

6.2 Additional Text Properties

These are additional attributes used by the IWB. Either the <iwb:element> attribute with a reference to either a <svg:text> or <svg:textarea> is used, or a <iwb:tspan> with a reference to a <svg:tspan>.

background-fill – The color of the background used behind the text.

Values: [<color> | none]

Initial: “none”.

Referenced to: svg:text, svg:textarea or svg:tspan.

highlight-fill - The color of the background used behind the text but used specifically to highlight the text.

Values: [<color> | none]

Initial: “none”.

Referenced to: svg:text, svg:textarea or svg:tspan.

type – The type the tspan shows, one of either normal or list.

Values: [normal | list]

Initial: “normal”.

Referenced to: svg:textarea or svg:tspan in svg:textarea.

list-style-type – The appearance of the list marker, e.g. circle or lower-alpha

Values: CSS list of “list-style-type”.

Initial: “circle”.

Referenced to: svg:textarea or svg:tspan in svg:textarea.

list-style-type-fill – The color of the list marker.

Values: [<color>]

Initial: “black”.

Referenced to: svg:textarea or svg:tspan in svg:textarea.

editable – Whether the text can be changed. (If the element has locked=”true” then the text is already not editable).

Values [true | false]

Initial: “true”

Referenced to: svg:text or svg:textarea.

Lists always start on a new line, with each further item starting on another new line. This example shows a list with three items:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:textarea font-family="Verdana" font-size="45" fill="black"
            x="10" y="60" width="100"
            height="200"> This is a list: <svg:tspan id="list1"> Item 1
                <svg:tbreak/> Item 2<svg:tbreak/> Item 3 </svg:tspan>
        </svg:textarea>
    </svg:svg>
    <tspan ref="list1" type="list" list-style-type="circle"/>
</iwb>
```

6.3 The Text Tags

6.3.1 Text

As in SVG the position of the baseline is set with the x and y co-ordinate, but unlike SVG, these are only single co-ordinates and therefore only specify the (x,y) position of the first character and not multiple characters. This can only represent a single line of text.

```
<svg:text x="10" y="10">A line of text.</svg:text>
```

6.3.2 Textarea

As in SVGT1.2 the textarea is defined as a rectangle on the page with x, y, width and height all in user co-ordinates. It should be noted that the wrapping of text may not appear exactly the same across different Readers, part of the text may wrap beyond the height of the box.

```
<svg:textarea x="0" y="0" width="50" height="90">
    This line can wrap onto several lines.
</svg:textarea>
```

6.3.3 Tbreak

As in SVG this is used to force a new line and is only available in a textarea.

```
<svg:textarea x="0" y="0" width="50" height="90">
    This line is broken<svg:tbreak/> into two lines.
</svg:textarea>
```

6.3.4 Tspan

This is used inside both <svg:text> and <svg:textarea> tags. Unlike SVG there is no positioning or offset changes here, the attributes rotate, x, y, dx and dy are not used.

```
<svg:text x="10" y="10" font-family="Verdana" font-size="45"
    fill="black">
    A <svg:tspan fill="red">line</svg:tspan> of
    <svg:tspan font-size="60" font-weight="bold">text</svg:tspan>.
</svg:text>
```

7 Elements: Media

Media includes images, audio, videos and flash. These are files embedded inside the zip version of the IWB file, placed inside their specific folder.

Images, videos and flash are elements in their own right, having a physical presence in the document. Audio is different in that it does not exist on its own but must be linked to another element.

Multiple references to the same file can be made inside the document.

7.1 Images

Images are stored in the folder called “images” in the zip file. The file types embedded are:

- Joint Photographic Experts Group, JPEG (*.jpeg; *.jpg)
- Bitmap (*.bmp)
- Animated and non animated Graphics Interchange Format, GIF (*.gif)
- Window Metafile and Enhanced Metafiles (*.wmf; *.emf)
- Portable Network Graphics, PNG (*.png)
- Tagged Image File Format, TIFF (*.tif;*.tiff)

Image elements are placed with the <svg:image> tag. They have an x and y position with a width and height. The image should stretch to fill this space. An xlink:href specifies which image it represents, the extension on the filename must remain intact so that file format is easily determined.

```
<svg:image xlink:href="images/myimage.png" x="10" y="10"
    width="100" height="100">
```

A requiredExtension attribute can also be added. The extension strings are of the type:

```
requiredExtension="http://www.imsglobal.org/iwb/wmf"
```

The string should end in the three letter id of the particular file format, the previous example has “wmf” to identify it as a “Windows Metafile”. If a <svg:switch> tag is not used to provide an alternative then no element will be displayed. See section “12 Fall back”.

7.1.1 Additional Image Properties

The IWB defines further attributes, added in the <iwb:element> tag.

flip – A reflection of the original image

Values: [none | horizontal | vertical | both].

Initial: “none”.

Referenced to: svg:image

7.2 Video

Videos are stored in the folder called “videos” in the zip file. The file types embedded should be:

Mpeg (*.mpg;*.mpeg)

Video elements are placed with the SVG1.2 tag <svg:video>. They have an x and y position with a width and height. The video should stretch to fill this space. An xlink:href specifies which video it represents.

```
<svg:video xlink:href="video/myvideo.mpeg" x="10" y="10"
    width="100" height="100">
```

A requiredExtension attribute can also be added. The extension strings are of the type:

```
requiredExtension="http://www.imsglobal.org/iwb/mpg"
```

The string should end in the three letter id of the particular file format, the previous example has “mpg” to identify it as a “MPEG video file”. If a <svg:switch> tag is not used to provide an alternative then no element will be displayed. See section“12 Fall back”.

7.3 Flash

Flash files are stored in the folder called “flash” in the zip file. The file types embedded should be:

Shockwave flash (*.swf)

Flash elements are also placed with the <svg:video> tag. They have an x and y position with a width and height. The flash should stretch to fit this space. An xlink:href specifies which flash it represents.

```
<svg:video xlink:href="flash/myflash.swf" x="10" y="10"
  width="100" height="100">
```

A requiredExtension attribute can also be added to the <svg:video> tag. The extension strings are of the type:

```
requiredExtension="http://www.imsglobal.org/iwb/swf"
```

The string should end in the three letter id of the particular file format, the previous example has “swf” to identify it as “Shockwave flash”. If a <svg:switch> tag is not used to provide an alternative then no element will display. See section “12 Fall back”.

7.4 Audio

Audio files are stored in the folder called “audio” in the zip file. The file types embedded should be:

Mp3 (*.mp3)

Wave (*.wav)

Audio files are not themselves elements and need to be linked to an element. An audio can be linked to any element except <svg:video>. See section “11 Links”. The link can also use section “12 Fall Back”. The requiredExtension strings for this are of the type:

```
requiredExtension="http://www.imsglobal.org/iwb/mp3"
```

The string should end in the three letter id of the particular file format, the previous example has “mp3” to identify it as a “MP3 Audio file”.

8 Groups

Groups are used to make several elements behave as if they are a single element. A group is defined with a <iwb:group> tag and surrounds two or more <iwb:element> tags. Any element can belong to a group, but note that the <svg:tspan> is not defined as an element in itself. No element can belong to more than one group but every group must have at least two elements.

To put two elements in to a group you would write:

```
<group>
    <element ref="rect1"/>
    <element ref="polyline1"/>
</group>
```

Additional IWB attributes can still be added to the individual elements, there is no need to define them twice. For instance, to mark the polyline as a freehand drawing you can add this to the element:

```
<group>
    <element ref="rect1" />
    <element ref="polyline1" freehand="true"/>
</group>
```

All elements in a group must exist on the same page, but the elements can be in any position on the page and on any z-order.

Groups should not be nested together, so this is illegal:

```
<group>
    <element ref="rect1" />
    <element ref="polyline1" />
    <group>
        <element ref="rect2" />
        <element ref="circle1" />
    </group>
</group>
```

9 Colors

Colors are represented in the same way as in SVG, which itself is based on the CSS2 definition. (See the CSS2 specification for more information, section 4.3.6 of that file)

Colors are used in the “fill” and “stroke” attributes.

There are five ways to represent color. Each of the following examples show color values representing a fill with color red and a stroke with color green.

Colors can be represented by a name, i.e. “name”. The IWB uses the SVG set of named colors.

```
<svg:rect x="0" y="0" width="10" height="10"
    fill="red" stroke="green"/>
```

Or they can be represented by a ‘#’ character followed by three hexadecimal characters, i.e. “#rgb”

```
<svg:rect x="0" y="0" width="10" height="10"
    fill="#f00" stroke="#0f0"/>
```

Or they can be represented by a ‘#’ character followed by six hexadecimal characters, i.e. “#rrggbb”

```
<svg:rect x="0" y="0" width="10" height="10"
    fill="#ff0000" stroke="#00ff00"/>
```

Or they can be represented by three decimal values from 0 to 255 enclosed inside “rgb()”, i.e. “rgb(red, green, blue)”

```
<svg:rect x="0" y="0" width="10" height="10"
    fill="rgb(255,0,0)" stroke="rgb(0,255,0)"/>
```

Or they can be represented by three percentages enclosed inside “rgb()”, i.e. “rgb(red%, green%, blue%)”

```
<svg:rect x="0" y="0" width="10" height="10"
    fill="rgb(100%,0%,0%)" stroke="rgb(0%,100%,0%)"/>
```

10 Background

You can set a background color and / or a background image to display.

10.1 Background Color

The background color is set with a <svg:rect> which must cover the viewbox area. It takes an attribute to identify it as a background. Only one rectangle can be used as a background and it must be placed first in a page's element list.

This example sets a page's color to red:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:pageset>
            <svg:rect id="rect1" fill="red" x="0" y="0" width="1000"
                height="1000"/>
            <svg:rect fill="white" x="450" y="450" width="100"
                height="100"/>
        </svg:pageset>
    </svg:svg>
    <element ref="rect1" background="true"/>
</iwb>
```

10.1.1 Additional Background Properties

The <iwb:element> tag can set this additional property:

background – States whether this element is a background.

Values: [true | false].

Initial: "false".

Referenced to: svg:rect or svg:image

10.2 Background Image

An image is set with the <svg:image> tag and an <iwb:element> tag. The <svg:image> tag, when used as a background, must be first on every page's element list, but immediately after the background rectangle if one is used.

A background image can either appear as a single image or as a tiled image repeated across the entire page.

By default the position and size attributes of the background <svg:image> tag are used in the same way as for a normal image, however, if the background-posture attribute is changed from "by-position" then the image is either stretched to fill the viewbox ("stretched-to-fill") , scaled to fit in the viewbox in its original aspect ("scaled-to-fit") or repeated across the entire page ("repeated"). Note that the image's position attributes SHOULD reflect the current background-posture setting except in the case of repeat. This example shows how an image is used to fill the viewbox area:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:image id="image1" xlink:href="images/myimage.png" x="0"
            y="0" width="1000" height="1000"/>
    </svg:svg>
    <element ref="image1" background="true"
        background-posture="stretched-to-fill"/>
</iwb>
```

For an image set to repeat, the position of the image is ignored and only the size is used, the image will repeat across the whole page not only the viewbox area. In this case the x and y attributes SHOULD be set to zero.

In this example an image is repeated on the background. The image will be repeated every 100 user co-ordinates across and every 100 user co-ordinates down.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:image id="image1" xlink:href="images/myimage.png"
            x="0" y="0" width="100" height="100"/>
    </svg:svg>
    <element ref="image1" background="true"
        background-posture="repeated"/>
</iwb>
```

The background image can also have a fall back image set, see section “12 Fall Back”. Each fall back image must have the background properties set.

10.2.1 Additional Background Image Properties

The <iwb:element> tag can set these additional properties:

background-posture – States how the image is displayed on the viewbox background. scaled to fit

Values: [scaled-to-fit | stretched-to-fill | repeated | by-position].

Initial: “by-position”.

Referenced to: svg:image

10.3 Combination

This example shows how to use a background color and a background image together. The rectangle must come immediately before the image so that transparent parts of the image will show the background color.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:rect id="background-color" fill="red" x="0"
            y="0" width="1000" height="1000"/>
        <svg:image id="background-image" xlink:href="images/myimage.png"
            x="450" y="450" width="100" height="100"/>
    </svg:svg>
    <element ref="background-color" background="true"/>
    <element ref="background-image" background="true"
        background-posture="repeated"/>
</iwb>
```

Background elements must not be contained in an iwb:group list.

11 Links

Links are achieved with the <svg:a> tag and an xlink:href attribute. If links appear somewhere in a file then the namespace for the xlink should be added to the file:

```
<iwb version="1.0" xmlns="http://www.imsglobal.org/xsd/iwb_v1p0"
  xmlns:svg="http://www.w3.org/2000/svg"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.imsglobal.org/xsd/iwb_v1p0
  http://www.imsglobal.org/profile/iwb/iwbv1p0_v1p0.xsd
  http://www.w3.org/2000/svg
  http://www.imsglobal.org/profile/iwb/svgsubsetv1p0_v1p0.xsd
  http://www.w3.org/1999/xlink
  http://www.imsglobal.org/xsd/w3/1999/xlink.xsd">
```

Links can be set on any of the elements or a selection of text inside a single element. For a link on a whole element, the element would be wrapped by the <svg:a> tag, this is an example using a rectangle:

```
<svg:a xlink:href="http://www.imsglobal.org">
    <svg:rect id="rect1" x="4" y="1" width="40" height="20"/>
</svg:a>
```

A link on a selection of text is achieved by wrapping the piece of text:

```
<svg:text x="10" y="10">
    A <svg:a xlink:href="http://www.imsglobal.org">link</svg:a> in some
    text.
</svg:text>
```

Multiple elements can be inside a single link tag if required as here:

```
<svg:a xlink:href="http://www.imsglobal.org">
    <svg:rect x="100" y="100" width="120" height="40"/>
    <svg:text x="130" y="100">Open website</svg:text>
</svg:a>
```

A link can be to the internal objects which include:

- an element on the current page
- another page
- an element on another page
- a media file embedded in the zip file

or to external objects which include:

- an external file
- a website address

11.1 Internal Object Links

Links to elements or pages are specified with a '#' followed by the id of the object. A link to an element on the same page causes the part of the page with that element on to be visible on screen, a link to another page means that the viewbox of that page is visible on screen, a link to an element on another page means that the part of the other page with that element on is visible on screen.

For instance, links to a rectangle from another rectangle works like so:

```
<svg:rect id="rect1" x="0" y="0" width="40" height="20" />
<svg:a xlink:href="#rect1">
    <svg:rect id="rect2" x="0" y="300" width="40" height="20" />
</svg:a>
```

11.2 Internal File Links

To link to an audio file embedded in the zip file you would give the URI relative to the xml file:

```
<svg:a xlink:href="audio/mytune.wav">
    <svg:text x="30" y="0">Play me<svg:text/>
</svg:a>
```

The <svg:a> tag can also take a requiredExtension attribute. If a <svg:switch> tag is not used to provide an alternative then no sound will play. See section “12 Fall back”.

11.3 External File Links

An external file, which is a file that exists outside of the zipped IWB file, can be linked to either with a relative or absolute URI. To tell apart an external file link from an internal one, external file links need an additional attribute. This is handled with a <iwb:link> tag which has file=”internal” or file=“external” setting. By default the links are internal.

This file opens the file “help.txt” located in the same directory as the IWB file:

```
<svg:svg width="800" height="600" viewbox="0 0 8000 6000">
    <svg:text x="30" y="0">
    Open <svg:a id="link1" xlink:href="help.txt">Help</svg:a>
    </svg:text>
</svg:svg>
<link ref="link1" file="external"/>
```

11.3.1 Additional External Links Attribute

Attributes for the <iwb:link> tag:

file – States whether the link is to an internal or external file

Values: [internal | external].

Initial: “internal”.

Referenced to: svg:a

11.4 Website Links

A website is linked to in a similar way:

```
<svg:a xlink:href="http://www.imsglobal.org">
    <svg:text x="30" y="0">The IMS website</svg:text>
</svg:a>
```

12 Fall Back

Some Readers may not be able to show all types of media so a fall back mechanism can be used so that an alternative can be used in its place. This is achieved with a combination of the SVG <svg:switch> tag and the requiredExtension attribute.

The switch surrounds the object that may need replacing. The object to be replaced must contain a requiredExtension attribute declaring what extension is needed to show the object, if this test fails then the next element between the <svg:switch> tags is used, if there is no next object then nothing will be displayed.

For instance, a document may contain a Window Metafile image but also include a portable network graphic image for Readers that can not display Metafiles. The file would look similar to this:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:switch>
            <svg:image x="0" y="0" width="10" height="10"
                xlink:href="images/myimage.wmf" xlink:type="simple"
                requiredExtension="http://www.imsglobal.org/iwb/wmf"/>
            <svg:image x="0" y="0" width="10" height="10"
                xlink:href="images/myimage.png" xlink:type="simple"/>
        </svg:switch>
    </svg:svg>
</iwb>
```

The above example has the problem that a Reader may not be able to display PNGs either, and possibly no image format, in this case any of the graphic or text elements could be used to fall back too.

There is also no limit to the number of fall back mechanism you can add to a switch. The switch will exit when it comes to an element that has requiredExtension a Reader can display.

This example has two possible fall backs for a metafile. If the metafile can't be displayed a bitmap is available, if that can't be displayed then a piece of text is displayed instead.

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:switch>
            <svg:image x="0" y="0" width="100" height="100"
                xlink:href="images/myimage.wmf" xlink:type="simple"
                requiredExtension="http://www.imsglobal.org/iwb/wmf"/>
            <svg:image x="0" y="0" width="10" height="10"
                xlink:href="images/myimage.bmp" xlink:type="simple"
                requiredExtension="http://www.imsglobal.org/iwb/bmp"/>
            <svg:text x="10" y="60" font-size="20"> Sorry, unable to
                display image. </svg:text>
        </svg:switch>
    </svg:svg>
</iwb>
```

The <svg:g> tag is also useful inside the <svg:switch> tag. Here, it can be used to enable several elements to replace another element. In this example if a metafile can't be displayed then both the rectangle and the text are displayed, without the <svg:g> tag only the rectangle would be shown:

```
<iwb>
    <svg:svg width="800" height="600" viewbox="0 0 1000 1000">
        <svg:switch>
            <svg:image x="0" y="0" width="50" height="50"
                xlink:href="image/myimage.wmf" xlink:type="simple"
                requiredExtension="http://www.imsglobal.org/iwb/wmf"/>
            <svg:g>
                <svg:rect x="0" y="0" width="50" height="50"/>
                <svg:text x="10" y="60" font-size="20">Unable to display
                    image.</svg:text>
            </svg:g>
        </svg:switch>
    </svg:svg>
</iwb>
```

Appendix A – Tag and Attribute Reference

This is a list of the tags and attributes that can appear in the IWB xml file.

A.1 IWB Tags

- element
- group
- iwb
- link
- meta
- tspan

element		Additional attributes for elements.
Attributes	ref	Compulsory. This holds the id of the SVG element it refers to.
	background	Optional. Determines if the referenced image or rectangle is used as a background.
	background-fill	Optional. The color of the background in the referenced textarea or text.
	background-posture	Optional. Determines how the referenced image is displayed on the background.
	flip	Optional. The orientation of the referenced image.
	freehand	Optional. States that the referenced polyline was created by freehand.
	highlight	Optional. States that the referenced polyline was created to highlight a part of a page.
	highlight-fill	Optional. The color of the background highlight in the referenced textarea or text.
	list-style-type	Optional. The appearance of the list marker in the referenced textarea.
	list-style-type-fill	Optional. The color of the list marker in the referenced textarea.
	locked	Optional. States whether the referenced element is locked to the page.
	replicate	Optional. States whether the referenced element should replicate itself rather than move.
	revealer	Optional. States whether the referenced rectangle is used to reveal an area of the page.
	stroke-lineshape-start	Optional. Adds a shape to the start of the referenced line or polyline.
	stroke-lineshape-end	Optional. Adds a shape to the end of the referenced line or polyline.
Example	*<iwb:element ref=”rect1” locked=”true” />*	
See also	The SVG elements (such as *svg:rect*, *svg:text* and *svg:image)*	

group		A group of elements
Sub tags	<iwb:element>	
Example	<iwb:group></iwb:group>	

iwb		This is the root element
Attributes	xmlns:iwb	Compulsory. The IWB namespace.
	xmlns:svg	Compulsory. The SVG namespace.
	xmlns:xlink	Compulsory with links. The XLINK namespace.
	version	Compulsory. The version of the IWB format that is being used.
Sub tags	*<iwb:element>, <iwb:group>, <iwb:link>, <iwb:meta>, <iwb:tspan>, <svg:svg>*	
Example	*<iwb xmlns:iwb="http://www.becta.org.uk/iwb" xmlns:svg="http://www.w3.org/2000/svg" version=”1.0”> </iwb>*	

link		A link to a file
Attributes	ref	Compulsory. This holds the id of the svg:a it refers to.
	file	Optional. States whether a link is to a file that is either internal or external to the zip.
Example	*<iwb:link ref=”link1” file=”external” />*	

meta		Extra data for file.
Attributes	name	The name of the extra data
	content	The extra data
Example	*<iwb:meta name=”owner” content=”John Smith” />*	

tspan	Additional attributes for tspans	
Attributes	ref	Compulsory. This holds the id of the SVG element it refers to.
	background-fill	Optional. States the background color of the tspan referenced.
	highlight-fill	Optional. States the background highlight color of the tspan referenced.
	list-style-type	Optional. Appearance of the list marker.
	list-style-type-fill	Optional. The color of the list marker.
	type	Optional. The type of the tspan referenced.
Example	*<iwb:tspan ref=”tspan1” type=”list” />*	
See also	*svg:tspan*	

A.2 SVG Tags

- svg:a
- svg:circle
- svg:ellipse
- svg:g
- svg:image
- svg:line
- svg:video
- svg:page
- svg:pageset
- svg:polygon
- svg:polyline
- svg:rect
- svg:svg
- svg:switch
- svg:tbreak
- svg:text
- svg:textarea
- svg:tspan

svg:a		Creates a link
Attributes	id	Compulsory if referenced. The id that identifies this link.
	xlink:href	Compulsory. The URI of the link.
Sub tags	*<svg:circle>, <svg:ellipse>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, svg:tbreak>, <svg:text>, <svg:textarea>, <svg:tspan>*	
Example	*<svg:a xlink:href=”http://schools.becta.org.uk/”> </svg:a>*	
See also	*<iwb:link>*	

svg:circle	Draws a circle on a page.	
Attributes	id	Compulsory if referenced. The id that identifies this circle.
	cx	Compulsory. The x position of the centercenter of the circle
	cy	Compulsory. The y position of the centercenter of the circle
	r	Compulsory. The radius of the circle.
	fill	Optional. Fill color of the circle.
	fill-opacity	Optional. Level of transparency for fill.

	stroke	Optional. The color for the circle edges.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the circle edge.
	stroke-linecap	Optional. How the ends of lines appear. Only valid if stroke-dasharray is also set.
	stroke-opacity	Optional. The level of transparency for the circle edge.
	stroke-width	Optional. The thickness of the circle edge.
	transform	Optional. A rotation and / or translation of the circle.
Example	*<svg:circle cx="50" cy="50" r="30" />*	
See also	*locked* and *replicate* attribute of *iwb:element*	

svg:ellipse		Draws an ellipse on a page.
Attributes	id	Compulsory. The id that identifies this ellipse.
	cx	Compulsory. The x position of the centercenter of the ellipse
	cy	Compulsory. The y position of the centercenter of the ellipse
	rx	Compulsory. The radius of the ellipse in the x direction.
	ry	Compulsory. The radius of the ellipse in the y direction.
	fill	Optional. Fill color of the ellipse.
	fill-opacity	Optional. Level of transparency for fill.
	stroke	Optional. The color for the ellipse edge.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the ellipse edge.
	stroke-linecap	Optional. How the ends of lines appear. Only valid if stroke-dasharray is also set.
	stroke-opacity	Optional. The level of transparency for the ellipse edge.
	stroke-width	Optional. The thickness of the ellipse edge.
	transform	Optional. A rotation and / or translation of the ellipse.
Example	*<svg:ellipse cx="700" cy="350" rx="350" ry="175"/>*	
See also	*locked* and *replicate* attribute of *iwb:element*	

svg:g		Container tag which holds other SVG elements. Attributes are applied to all contained elements if that particular attribute is valid for a particular element.
Attributes	fill	Optional. Fill color of any shape or text element contained.
	fill-opacity	Optional. Level of transparency for fill.
	stroke	Optional. The line color of any line or shape contained.
	stroke-	Optional. A comma separated list of dash and gap lengths

	dasharray	representing a stroke pattern.
	stroke-linecap	Optional. How the ends of lines appear.
	stroke-linejoin	Optional. How line joins appear.
	stroke-opacity	Optional. The level of transparency for lines and shape edges contained.
	stroke-width	Optional. The thickness of any lines and shape edges contained.
	transform	Optional. A rotation and / or translation of all the elements contained.
Sub tags	*<svg:a>, <svg:circle>, <svg:ellipse>, <svg:g>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, <svg:switch>, <svg:text>, <svg:textarea>*	
Example	*<svg:g fill=”red” transform=”rotate(10)”> <!-- SVG elements here --> </svg:g>*	

svg:image		An image placed on the page.
Attributes	id	Compulsory if referenced. The id that identifies this image.
	xlink:href	Compulsory. The internal URL to the image.
	x	Compulsory. The left position of the image.
	y	Compulsory. The top position of the image.
	width	Compulsory. The width of the image.
	height	Compulsory. The height of the image.
	fill-opacity	Optional. Level of transparency for this image.
	requiredExtension	Optional. The extension that is required by a Reader to display this image.
	transform	Optional. A rotation and / or translation of the image.
Example	*<svg:image xlink:href=”images/enterprise.png” x=”0” y=”0” width=”300” height=”300” />*	
See also	*flip, locked* and *replicate* attribute of *iwb:element*	

svg:line	Draws a line on a page.	
Attributes	id	Compulsory if referenced. The id that identifies this line.
	x1	Compulsory. The x position of the start of the line.
	y1	Compulsory. The y position of the start of the line.
	x2	Compulsory. The x position of the end of the line.
	y2	Compulsory. The y position of the end of the line.
	stroke	Optional. The color for the line.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the line.
	stroke-linecap	Optional. How the ends of the line appear.

	stroke-opacity	Optional. The level of transparency for the line.
	stroke-width	Optional. The thickness of the line.
	transform	Optional. A rotation and / or translation of the line.
Example	*<svg:rect x1="150" y1="50" x2="100" y2="50" />*	
See also	*stroke-lineshape-start, stroke-lineshape-end, locked* and *replicate* attribute of *iwb:element*	

svg:video		A video (or flash) on a page.
Attributes	id	Compulsory if referenced. The id that identifies this video.
	xlink:href	Compulsory. The internal URL to the video.
	x	Compulsory. The left position of the video.
	y	Compulsory. The top position of the video.
	width	Compulsory. The width of the video.
	height	Compulsory. The height of the video.
	requiredExtension	Optional. The extension that is required by a Reader to display this video.
	transform	Optional. A rotation and / or translation of the video.
Example	*<svg:video xlink:href="videos/tardis.mpeg" x="0" y="0" width="300" height="300" />*	
See also	*flip, locked* and *replicate* attribute of *iwb:element*	

svg:page		Start a page.
Attributes	id	Compulsory if linked to. The id to identify this page. Used in Linking.
Sub tags	*<svg:a>, <svg:circle>, <svg:ellipse>, <svg:g>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, <svg:switch>, <svg:text>, <svg:textarea>*	
Example	*<svg:page id="page1"> </svg:page>*	
See also	*<svg:pageset>*	

svg:pageset	Start a collection of one or more pages.
Sub tags	*<svg:page>*
Example	*<svg:pageset> </svg:pageset>*

svg:polygon		Draws a polygon on a page.
Attributes	id	Compulsory if referenced. The id that identifies this polygon.
	points	Compulsory. A list of x and y points representing the edge of the

		shape.
	fill	Optional. Fill color of the rectangle.
	fill-opacity	Optional. Level of transparency for fill.
	stroke	Optional. The color for the rectangle edges.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the polygon edge.
	stroke-linecap	Optional. How the ends of lines appear. Only valid if stroke-dasharray is also set.
	stroke-linejoin	Optional. How the joins in the edges appear.
	stroke-opacity	Optional. The level of transparency for the polygon edges.
	stroke-width	Optional. The thickness of the polygon edges.
	transform	Optional. A rotation and / or translation of the polygon.
Example	*<svg:polygon points="50,375 150,375 150,325 250,325" />*	
See also	*locked* and *replicate* attribute of *iwb:element*	

svg:polyline		Draws a polyline on a page.
Attributes	id	Compulsory if referenced. The id that identifies this polyline.
	points	Compulsory. A list of x and y points representing one or more lines joined together.
	stroke	Optional. The color for the polyline.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the polyline.
	stroke-linecap	Optional. How the ends of the polyline appear.
	stroke-opacity	Optional. The level of transparency for the polyline.
	stroke-width	Optional. The thickness of the polyline.
	transform	Optional. A rotation and / or translation of the polyline.
Example	*<svg:polyline points="50,375 150,375 150,325 250,325" />*	
See also	*stroke-lineshape-start, stroke-lineshape-end, freehand, highlight, locked,* and *replicate* attribute of *iwb:element*	

svg:rect	Draws a rectangle on a page	
Attributes	id	Compulsory if referenced. The id that identifies this rectangle.
	x	Compulsory. The left position of the rectangle.
	y	Compulsory. The top position of the rectangle.
	width	Compulsory. The width of the rectangle.
	height	Compulsory. The height of the rectangle.
	fill	Optional. Fill color of the rectangle.

	fill-opacity	Optional. Level of transparency for fill.
	stroke	Optional. The color for the rectangle edges.
	stroke-dasharray	Optional. A comma separated list of dash and gap lengths representing a pattern for the shape edge.
	stroke-linecap	Optional. How the ends of lines appear. Only valid if stroke-dasharray is also set.
	stroke-linejoin	Optional. How the corners of the rectangle appear.
	stroke-opacity	Optional. The level of transparency for the rectangle edges.
	stroke-width	Optional. The thickness of the rectangle edges.
	transform	Optional. A rotation and / or translation of the rectangle.
Example	*<svg:rect x="0" y="0" width="300" height="300" />*	
See also	*revealer*, *locked* and *replicate* attribute of *iwb:element*	

svg:svg		Start the SVG part of the file.
Attributes	viewbox	Compulsory. The area on a page that should be shown as a "slide".
	width	Optional. The original width of the screen in pixels.
	height	Optional. The original height of the screen in pixels.
Sub tags	*<svg:a>*, *<svg:circle>*, *<svg:ellipse>*, *<svg:g>*, *<svg:image>*, *<svg:line>*, *<svg:video>*, *<svg:pageset>*, *<svg:polygon>*, *<svg:polyline>*, *<svg:rect>*, *<svg:switch>*, *<svg:text>*, *<svg:textarea>*	
Example	*<svg:svg width="800" height="600" viewbox="0 0 8000 6000"> </svg:svg>*	

svg:switch	A switch so elements can be chosen from a selection.
Sub tags	*<svg:circle>*, *<svg:ellipse>*, *<svg:g>*, *<svg:image>*, *<svg:line>*, *<svg:video>*, *<svg:polygon>*, *<svg:polyline>*, *<svg:rect>*, *<svg:text>*, *<svg:textarea>*
Example	*<svg:switch> </svg:switch>*

svg:tbreak	Adds a new line inside a textarea
Example	*<svg:textarea x="0" y="0" width="100" height="50">Hello<tbreak />world!</svg:textarea>*
See also	*svg:textarea*

svg:text		A single line of text.
Attributes	id	Compulsory if referenced. The id to identify the text.
	X	Compulsory. The x position of the text baseline.
	Y	Compulsory. The y position of the text baseline.
	Fill	Optional. The default text color for this text.
	Font-family	Optional. The default font for this text.

	Font-size	Optional. The default text size for this text.
	Font-style	Optional. The default style for this text.
	Font-weight	Optional. The default weight for this text.
	Font-stretch	Optional. The default horizontal only stretch for this text.
	Transform	Optional. A rotation and / or translation of the text.
Sub tags	*<svg:tspan>*	
Example	*<svg:text x="30" y="10" font-size="20">Hello world!</svg:text>*	
See also	*iwb:tspan*	

svg:textarea		A single line of text.
Attributes	id	Compulsory if referenced. The id to identify the textarea.
	x	Compulsory. The left edge of the textarea.
	y	Compulsory. The top edge of the textarea.
	width	Compulsory. The width of the textarea. Text automatically wraps to the next line here.
	height	Compulsory. The height of the textarea.
	fill	Optional. The default text color for this textarea.
	font-family	Optional. The default font for this textarea.
	font-size	Optional. The default text size for this textarea.
	font-stretch	Optional. The default horizontal only stretch for this textarea.
	font-style	Optional. The default style for this textarea.
	font-weight	Optional. The default weight for this textarea.
	text-align	Optional. The default justification for this textarea.
	transform	Optional. A rotation and / or translation of the textarea.
Sub tags	*<svg:tbreak>, <svg:tspan>*	
Example	*<svg:textarea x="0" y="0" width="100" height="50" fill="red">Hi Earthlings!</svg:textarea>*	
See also	*iwb:tspan*	

svg:tspan		A single line of text.
Attributes	id	Compulsory if referenced. The id to identify this tspan.
	fill	Optional. The color of the text in the tspan.
	font-family	Optional. The font of the text in the tspan.
	font-size	Optional. The size of the text in the tspan.
	font-stretch	Optional. The horizontal only stretch of the text in the tspan.

	font-style	Optional. The style of the text in the tspan.
	font-weight	Optional. The weight of the text in the tspan.
	text-align	Optional. The justification of the text in the tspan. Only for use when a child of a textarea.
Sub tags	*<svg:tbreak>, <svg:tspan>*	
Example	*<svg:text x="50" y="0" fill="black">Greetings <tspan fill="green">Gaia</tspan>!</svg:text>*	
See also	*svg:text, svg:textarea* and *iwb:tspan*	

Appendix B – Standard "Core" or "Full" Set

The Core set of tags and attributes is the minimal set that an application must support to be certified IWB compatible. The Full set is a superset of the Core set and includes all the attributes and tags of the specification.

Attributes not handled by a Reader can be skipped over and ignored, although depending on what is skipped, a User may need to be informed if the result is that the file is drastically different from what is stored in the original file.

Attributes not listed here explicitly are part of the Core set.

Tag	Attributes	Core	Full
element		Core	
	background	Core	
	background-fill		Full
	background-posture		Full
	flip	Core	
	freehand		Full
	highlight		Full
	highlight-fill		Full
	list-style-type		Full
	list-style-type-fill		Full
	locked	Core	
	replicate		Full
	revealer		Full
	stroke-lineshape-start		Full
	stroke-lineshape-end		Full
group		Core	
iwb		Core	
	version	Core	
	xmlns	Core	
	xmlns:svg	Core	
	xmlns:xlink	Core	
	xmlns:xsi	Core	
	xsi:schemaLocation	Core	
link		Core	
meta		Core	
tspan		Core	
	background-fill		Full
	highlight-fill		Full

	list-style-type		Full
	list-style-type-fill		Full
	type	Core	
svg:a		Core	
	xlink:href	Core	
	xlink:href=”*.wav”	Core	
	xlink:href=”*.mp3”		Full
svg:circle		Core	
	fill	Core	
	fill-opacity	Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	
svg:ellipse		Core	
	fill	Core	
	fill-opacity	Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	
svg:g		Core	
	fill	Core	
	fill-opacity	Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke_linejoin		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	

svg:image		Core	
	xlink:href=”*.jpg”	Core	
	xlink:href=”*.bmp”	Core	
	xlink:href=”*.gif”	Core	
	xlink:href=”*.png”	Core	
	xlink:href=”*.wmf”		Full
	xlink:href=”*.emf”		Full
	xlink:href=”*.tif”		Full
	fill-opacity		Full
	requiredExtension	Core	
	transform	Core	
svg:line		Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	
svg:video		Core	
	xlink:href=”*.swf”		Full
	xlink:href=”*.mpeg”	Core	
	requiredExtension	Core	
	transform	Core	
svg:page		Core	
svg:pageset		Core	
svg:polygon		Core	
	fill	Core	
	fill-opacity	Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke_linejoin		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	

svg:polyline		Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	
svg:rect		Core	
	fill	Core	
	fill-opacity	Core	
	stroke	Core	
	stroke-dasharray		Full
	stroke-linecap		Full
	stroke_linejoin		Full
	stroke-opacity		Full
	stroke-width	Core	
	transform	Core	
svg:svg		Core	
	viewbox	Core	
	height		Full
	width		Full
svg:switch		Core	
svg:tbreak		Core	
svg:text		Core	
	fill	Core	
	font-family	Core	
	font-size	Core	
	font-style	Core	
	font-weight	Core	
	font-stretch		Full
	transform	Core	
svg:textarea		Core	
	fill	Core	
	font-family	Core	
	font-size	Core	

	font-stretch		Full
	font-style	Core	
	font-weight	Core	
	text-align	Core	
	transform	Core	
svg:tspan		Core	
	fill	Core	
	font-family	Core	
	font-size	Core	
	font-stretch		Full
	font-style	Core	
	font-weight	Core	
	text-align	Core	

Appendix C – Changes from Becta Document to IMS 1.0

The changes from the original Becta Interactive Whiteboard Common File Format to the IMS IWB specification version 1.0 were designed to be minimal. The changes are mostly comprised of XML Schema and Schematron rules the IMS depends upon to assist vendors in gaining IMS compliance certification. Most Becta files can be made compliant by changing the content.xml file to be more explicit about what namespaces it uses and where the actual XML schema files reside. A typical original Becta file would need to have the following changes made to bring it into compliance with the IMS IWB 1.0 specification:

1. Replace the Becta namespace declaration (xmlns:iwb=http://www.becta.org.uk/iwb) with the IMS IWB namespace (xmlns=http://www.imsglobal.org/xsd/iwb_v1p0).
2. Declare the IMS IWB namespace as the default namespace for the IWB elements and remove the "iwb:" namespace prefix from those IWB elements.
3. Use the schemaLocation attribute to specifically declare all of the namespaces and their document locations in each content.xml XML instance document like this:

```
xsi:schemaLocation="http://www.imsglobal.org/xsd/iwb_v1p0
http://www.imsglobal.org/profile/iwb/iwbv1p0_v1p0.xsd http://www.w3.org/2000/svg
http://www.imsglobal.org/profile/iwb/svgsubsetv1p0_v1p0.xsd http://www.w3.org/1999/xlink
http://www.imsglobal.org/xsd/w3/1999/xlink.xsd">
```

About This Document

Title:	IMS GLC IWB/CFF Conformance Document
Editor:	Thor Anderson (Utah Valley University)
Version:	1.0
Version Date:	1 February 2012
Release:	1.0
Status:	**Final Release**
Summary:	This document describes the IWB/CFF specification.
Purpose:	This document is made available for adoption by the IMS community and general public.
Document Location:	Join the discussion and post comments in the IWB/CFF Forum: http://www.imsglobal.org/community/forum/categories.cfm?catid=145&entercat=y

List of Contributors

The following individuals contributed to the development of this document:

Name	Organization
Dinesh Advani	SMART Technologies
Thor Anderson	Utah Valley University
Martin Hall	Lightbox Education
Lisa Mattson	IMS Global Learning Consortium
Jordan Meyerowitz	SMART Technologies
Jitu Patel	TeamBoard
Colin Smythe	IMS Global Learning Consortium
Mary Waller	TeamBoard

Revision History

Version No.	Release Date	Comments
Public Draft v1.0	7 December 2011	This draft is submitted to the IMS community for public review and comment.
Final Release v1.0	1 February 2012	This specification is formally approved by IMS GLC for adoption and conformance.